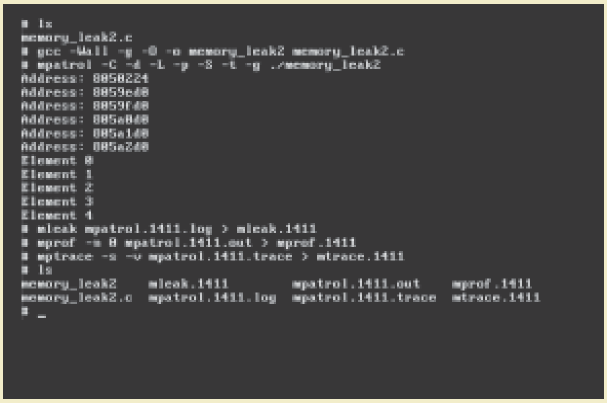
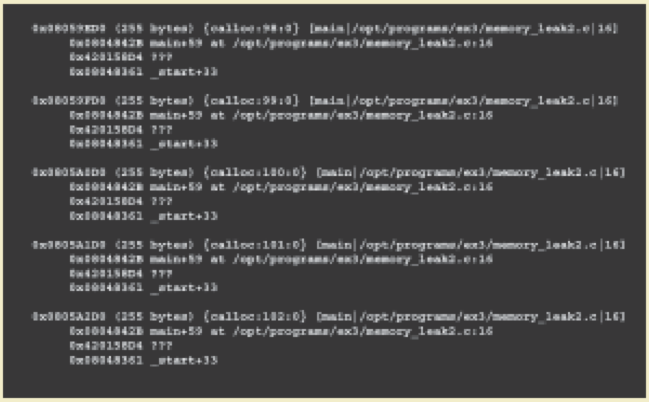


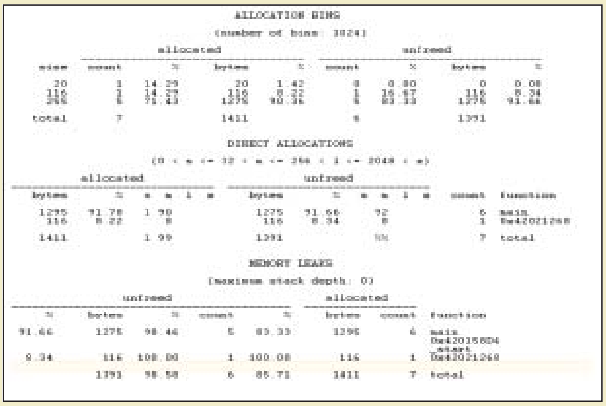
```
# ls
memory_leak2.c
# gcc -Wall -g -O -o memory_leak2 memory_leak2.c
# mpatrol -C -d -L -p -S -t -g ./memory_leak2
Address: 08058224
Address: 0859ed00
Address: 0859fd00
Address: 085a0d00
Address: 085a1d00
Address: 085a2d00
Element 0
Element 1
Element 2
Element 3
Element 4
# mleak mpatrol.1411.log > mleak.1411
# mprof -n 0 mpatrol.1411.out > mprof.1411
# mptrace -s -v mpatrol.1411.trace > mtrace.1411
# ls
memory_leak2    mleak.1411          mpatrol.1411.out    mprof.1411
memory_leak2.c  mpatrol.1411.log    mpatrol.1411.trace  mtrace.1411
# _
```

Figure 4



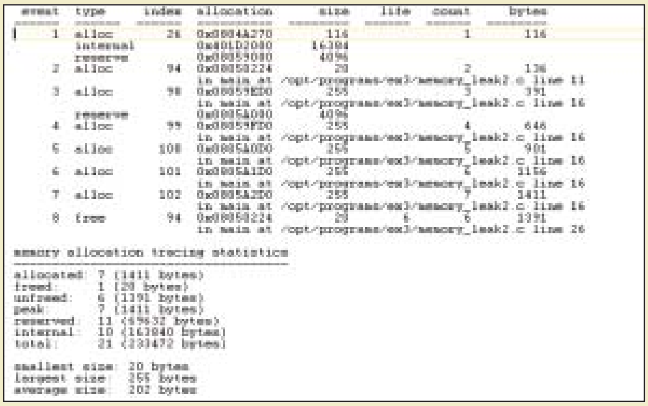
```
0x08059ED0 (255 bytes) {calloc:98:0} [main|/opt/programs/ex3/memory_leak2.c|16]
      0x0804842B main+59 at /opt/programs/ex3/memory_leak2.c:16
      0x420158D4 ???
      0x08048361 _start+33

0x08059FD0 (255 bytes) {calloc:99:0} [main|/opt/programs/ex3/memory_leak2.c|16]
      0x0804842B main+59 at /opt/programs/ex3/memory_leak2.c:16
      0x420158D4 ???
      0x08048361 _start+33

0x0805A0D0 (255 bytes) {calloc:100:0} [main|/opt/programs/ex3/memory_leak2.c|16]
      0x0804842B main+59 at /opt/programs/ex3/memory_leak2.c:16
      0x420158D4 ???
      0x08048361 _start+33

0x0805A1D0 (255 bytes) {calloc:101:0} [main|/opt/programs/ex3/memory_leak2.c|16]
      0x0804842B main+59 at /opt/programs/ex3/memory_leak2.c:16
      0x420158D4 ???
      0x08048361 _start+33

0x0805A2D0 (255 bytes) {calloc:102:0} [main|/opt/programs/ex3/memory_leak2.c|16]
      0x0804842B main+59 at /opt/programs/ex3/memory_leak2.c:16
      0x420158D4 ???
      0x08048361 _start+33
```

Figure 5



```
                          ALLOCATION BINS
                       (number of bins: 1024)
            allocated                              unfreed
     --------------------------------   --------------------------------
size   count     %     bytes     %     count     %     bytes     %

  20       1   14.29      20    1.42       0    0.00       0    0.00
 116       1   14.29     116    8.22       1   16.67     116    8.34
 255       5   71.43    1275   90.36       5   83.33    1275   91.66

total      7            1411               6            1391

                         DIRECT ALLOCATIONS
            (0 < s <= 32 < m <= 256 < l <= 2048 < x)
        allocated                        unfreed
 ------------------------------   ------------------------------
 bytes     %     s    m   l  x    bytes     %     s    m   l  x    count  function

  1295   91.78   1   90            1275   91.66        92            6  main
   116    8.22        8             116    8.34         8            1  0x42021268

  1411           1   99            1391                %%            7  total

                           MEMORY LEAKS
                     (maximum stack depth: 0)
               unfreed                         allocated
 ---------------------------------------   -----------------
    %    bytes     %    count     %        bytes    count  function

 91.66    1275   98.46      5   83.33      1295        6  main
                                                          0x420158D4
                                                          _start
  8.34     116  100.00      1  100.00       116        1  0x42021268

          1391   98.58      6   85.71      1411        7  total
```

Figure 6



```
 event  type      index  allocation        size    life   count    bytes
 -----  --------  -----  ----------  ----------  ------  ------  -------
     1  alloc        26  0x0804A270         116               1      116
        internal         0x401D2000       16384
        reserve          0x08059000        4096
     2  alloc        94  0x08058224          20               2      136
                         in main at /opt/programs/ex3/memory_leak2.c line 11
     3  alloc        98  0x08059ED0         255               3      391
                         in main at /opt/programs/ex3/memory_leak2.c line 16
        reserve          0x0805A000        4096
     4  alloc        99  0x08059FD0         255               4      646
                         in main at /opt/programs/ex3/memory_leak2.c line 16
     5  alloc       100  0x0805A0D0         255               5      901
                         in main at /opt/programs/ex3/memory_leak2.c line 16
     6  alloc       101  0x0805A1D0         255               6     1156
                         in main at /opt/programs/ex3/memory_leak2.c line 16
     7  alloc       102  0x0805A2D0         255               7     1411
                         in main at /opt/programs/ex3/memory_leak2.c line 16
     8  free         94  0x08058224          20       6       6     1391
                         in main at /opt/programs/ex3/memory_leak2.c line 26

memory allocation tracing statistics
------------------------------------
allocated:  7 (1411 bytes)
freed:      1 (20 bytes)
unfreed:    6 (1391 bytes)
peak:       7 (1411 bytes)
reserved:   11 (69632 bytes)
internal:   10 (163840 bytes)
total:      21 (233472 bytes)

smallest size:  20 bytes
largest size:   255 bytes
average size:   202 bytes
```

Figure 7

even if your program is not linked with *mpatrol* library. *mpatrol* does this by preloading the *mpatrol* library before loading your program.

The '-L' option instructs *mpatrol* to log all *allocs*, *reallocs* and *deallocs* that your program performs. The '-p' option instructs *mpatrol* to enable memory allocation profiling. The '-S' option instructs *mpatrol* to show the summary of free, freed and unfreed memory blocks/allocations, memory map and program symbols at the end of your program execution. The '-g' option instructs *mpatrol* to use debugging information, if any, from your program to log more source-level details. The '-t' option instructs *mpatrol* to enable memory tracing.

mpatrol in action

You will observe that outputs in Figures 6 and 7 also show some additional memory being allocated, but not freed. You have to remember that *mpatrol* not only profiles your program, but also the libraries that were linked with your program.

The *mleak* output shown in Figure 5, shows the details of the memory that has been allocated by the program, but not recovered. The output shows that the memory allocated at line number 16 in memory_leak2.c has not been freed. The *mprof* output shown in Figure 6 represents the memory allocation profile of the program. It gives the total amount of memory that has been allocated and not freed by the program. The *mtrace* output shown in Figure 7 presents the memory tracing statistics of the program. The report shows the amount of memory that has been allocated and freed by the program by line number.

As these reports clearly demonstrate, tools like *mpatrol* can help you detect the memory leaks in your program and so make your code more reliable than ever before. LFY

By: Raja R.K. The author is a lead engineer working in HCL Technologies (Cisco Systems Offshore Development Centre) in Chennai. He has more than five years of experience in software development including two years in Linux.